Deriving business processes with service level agreements from early requirements

Frankova, Ganna; Seguran, Magali; Gilcher, Florian; Trabelsi, Slim; Doerflinger, Joerg; Aiello, Marco

Published in:
Journal of Systems and Software

DOI:
10.1016/j.jss.2011.03.077

Publication date:
2011

The Journal of Systems and Software 84 (2011) 1351–1363

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

Deriving business processes with service level agreements from early requirements

Ganna Frankova[a,*], Magali Séguran[b], Florian Gilcher[b,1], Slim Trabelsi[b], Jörg Dörflinger[c], Marco Aiello[d]

[a] Department of Information Engineering and Computer Science, University of Trento, Via Sommarive, 14, 38050 Trento, Italy
[b] SAP Labs France, SAP Research – Security and Trust, 805, avenue du Dr. Maurice Donat, 06254 Mougins Cedex, France
[c] SAP Labs Germany, SAP Research, CEC Karlsruhe, Vincenz-Priessnitz-Straße 1, 76131 Karlsruhe, Germany
[d] Johann Bernoulli Institute, University of Groningen, Nijenborgh 9, 9747 AG Groningen, The Netherlands

ARTICLE INFO

Article history:
Received 14 October 2009
Received in revised form 26 January 2011
Accepted 29 March 2011
Available online 8 April 2011

Keywords:
Business process
Service level agreement
Requirements engineering

ABSTRACT

When designing a service-based business process employing loosely coupled services, one is not only interested in guaranteeing a certain flow of work, but also in how the work will be performed. This involves the consideration of non-functional properties which go from execution time and costs, to trust and security. Ideally, a designer would like to have guarantees over the behavior of the services involved in the process. These guarantees are the object of Service Level Agreements.

We propose a methodology to design service-based business processes together with Service Level Agreements that guarantee a certain quality of execution, with particular emphasis on security. Starting from an early requirements analysis modeled in the Secure Tropos formalism, we provide a set of user-guided transformations and reasoning tools the final output of which is a set of processes in the form of Secure BPELs together with a set of Service Level Agreements to be signed by participating services. To show the potential impact of the approach, we illustrate the functioning of the methodology on a collaborative procurement scenario derived from the application domain of a research project.

1. Introduction

Web services features of autonomy, platform-independence, readiness to be described, published, discovered, and orchestrated are increasingly exploited by companies to build massively distributed and loosely coupled interoperable applications (Papazoglou and Georgakopoulos, 2003). Enterprises not only export their services as Web services, but also develop their Business Process (BP) to be Web service-based. Since services may be offered by different providers, non-functional properties become of a paramount importance in defining the usability and success both of a service and of Web service-based BP.

The non-functional properties of a service can be agreed a priori between a Web service provider and a consumer by specifying a Service Level Agreement (SLA) (Molina-Jimenez et al., 2005; Andrieux et al., 2007; Ludwig et al., 2003). SLA for Web services is a binding contract between a Web service provider and a consumer which specifies a collection of service level requirements that are negotiated and mutually agreed upon (Cappiello et al., 2007). In Aiello et al. (2005) and Frankova et al. (2006), we provide semantics of an SLA protocol for Web services considering each SLA term as containing a guarantee, i.e., a right or obligation of the signing parties. The objects of an SLA can take many forms: it can be a maximum response time, a cost per operation, or it can take more complex forms such as "the service should execute in less than 5 s 99% of the times from 9 a.m. to 5 p.m.". Without loss of generality, we concentrate on the first case in this paper.

Service Level Agreements are thus an important tool for guiding the quality of the execution of business processes, though, their definition is not straightforward from the business goals of the organization running the processes (How to Series, 2005). One might say that there is a natural tension between implementing SLAs and their relation to the actual business goals of the organization. With the present work, we try to ease such tension by providing a methodology to derive executable processes from business requirements. The methodology, named BP&SLA, sticks out for a number of innovative features such as its starting from informally specified early requirements and the fact that the final products of the methodology are executable Web service-based business processes together with automatically negotiable Service Level Agreements. In fact, the output hierarchy of BPs is expressed in terms of the standards WS-BPEL for the processes

* Corresponding author.
E-mail addresses: frankova@disi.unitn.it, gannafrankova@yahoo.com (G. Frankova), magali.seguran@sap.com (M. Séguran), florian.gilcher@sap.com, flo@andersground.net (F. Gilcher), slim.trabelsi@sap.com (S. Trabelsi), joerg.doerflinger@sap.com (J. Dörflinger), aiellom@cs.rug.nl (M. Aiello).
[1] Present address: Gartenstraß e 31b, D-67105 Schifferstadt, Germany.

0164-1212/$ – see front matter © 2011 Elsevier Inc. All rights reserved.
doi:10.1016/j.jss.2011.03.077

and WS-Agreement for the SLAs. The methodology rationalizes and builds upon our previous results in providing a semantics for WS-Agreement (Frankova et al., 2006) and defining an extension of WS-BPEL to express security properties which can be derived from early requirements (Frankova et al., 2007a; Sèguran et al., 2008). This paper provides, for the first time, the full methodology from early requirements to executable processes and SLAs and includes a description of an implementation based on the ECLiPSe constraint satisfaction environment.

To illustrate the methodology, we consider a business process of a typical collaborative procurement scenario coming from an actual African project of SAP. The scenario includes service providers as stores, service entrepreneurs that provide information and data related to services in rural areas, warehouses and logistic partners. Each of the actors provides several services such as ordering, managing payment, products packaging, and distribution, to the end-consumer or to another provider. As in the area of procurement, both quality of execution and security of a process are crucial; each service is assigned a set of service level requirements negotiated between the provider and the consumer.

The remainder of the paper is organized as follows. Related work is discussed in Section 2. In Section 3, we introduce the collaborative procurement use case in the Sekhukhne Rural Living Lab case study that is used as a running example throughout the paper. Section 4 is devoted to the proposed BP&SLA methodology, which is applied to the running example in Section 5. The description of the implementation is presented in Section 6. Concluding remarks are summarized in Section 7. The two central algorithms of the methodology are reported in Appendices A and B.

2. Background and related work

Requirements engineering methodologies for business process creation and management has been gaining increasing attention both in the Software Engineering and in the Service-Oriented communities. Next, we provide an overview of the most representative approaches. We begin by looking at Service Level Agreement and a description of the approach aimed to its formal specification and negotiation, then we review trust and reputation works. Finally, we present the requirements engineering methodologies in the context of Web service design.

2.1. Service level agreement

In order to be reliable and thus successful, Web service providers have to offer and meet guarantees related to the services they offer. Taking into account that a guarantee depends on actual resource usage, a service consumer must request state-dependent guarantees from a service provider. Additionally, the guarantees on service quality must be monitored and service consumers must be notified in case of failure of meeting the guarantees. An agreement between a service consumer and a service provider, i.e., Service Level Agreement specifies the associated guarantees. The agreement can be formally specified using the WS-Agreement Specification (Andrieux et al., 2007). The WS-Agreement protocol proposal is supported by the definition of a managing architecture: CREMONA – An Architecture and Library for Creation and Monitoring of WS-Agreement (Ludwig et al., 2004). The Web Services Agreement Specification defines the interaction between a service provider and a consumer, and a protocol for creating an agreement using agreement templates. Aiello, Frankova and Malfatti in Aiello et al. (2005) provide a formal definition of what the semantics of a QoS negotiation should be and extend the framework in order to enhance flexibility at execution time. The issue of SLA generation we touch upon in the present work, is not well developed. An approach proposed by Cappiello et al. (2007) presents a negotiation model to support the automatic generation of SLA on-the-fly. The authors develop a model to express Web service quality, provider capabilities, and user requirements that is further employed in the negotiation model to generate SLA. In our approach, we tie BPs with SLAs. We do not focus on SLA negotiation, while we take into account early requirements provided by the BP user, the structure of the BP and security and trust concepts.

2.2. Trust and reputation

Trust is a directed relationship between two parties that can be called the trustor and the trustee. Trust is an essential aspect for decision on security since it is related to belief in honesty, competence and reliability (Castelfranchi and Falcone, 1998; McKnight and Chervany, 1996). Trust is not symmetric, so this belief by the trustor does not necessarily imply any similar belief by the trustee. Distrust is a quantified belief by a trustor that a trustee is incompetent, dishonest, not secure or not dependable within a specified context. In Gambetta (1990), Gambetta emphasizes the subjective level of trust: "trust is the subjective probability by which an individual A, expects than another individual, B, performs a given action on which its welfare depends". In Castelfranchi and Falcone (1998), Castelfranchi and Falcone consider trust from a cognitive point of view. They argue that it is a mental state based on a set of beliefs (depending on the feeling of trust more than the trust itself). There are various reasons for distrusting agents such as unskillfulness, unreliability and abuse. According to the authors, trust implies that having high trust in a person is not sufficient to imply the decision of trust, it could depend on the situation and the evaluation of the risk (Falcone and Castelfranchi, 2001). Usually, there is a level of trust associated with a trust relationship. Trustworthiness is defined as a measure of the level of trust that the trusting agent has in the trusted agent. The trust level is a measure of belief in another entity and thus it is a measure of belief in the honesty, competence, security and dependability of this entity (not a measure of the actual competence, honesty, security or dependability of a trustee) (Grandison and Sloman, 2003). Considering the trust level, we emphasize the following two approaches. First, there might be some degrees in the trust level, i.e., the so called "[0 . . . 1] trust level approach" that points the level of trust of one entity to another one. It indicates some degree between the absence and the presence of trust. In the definition given in Grandison and Sloman (2002) and Grandison and Sloman (2003), quantification is linked to the notion of trust. Quantification reflects that a trustor can have various degrees of trust (distrust), which could be expressed as a numerical range or as a discrete classification such as low, medium or high (Grandison and Sloman, 2003). The work done in SULTAN (Grandison and Sloman, 2002, 2003) (Simple Universal Logic-oriented Trust Analysis Notation) has incorporated concepts such as experience, reputation and trusting propensity. One of the disadvantages of the "[0 . . . 1] trust level approach" is that it is not clear how to define the exact degree of the trust level.

Second, the "0/1 trust level approach" means a strict absence/presence of trust dependencies. In Asnar et al. (2007), the authors consider three trust levels: Trust, Distrust, and NTrust (i.e., neither trust nor distrust). Trust and Distrust means 1/0 trust level, NTrust is necessary since the requirements specification may not define any trust or distrust relation between two specific actors. In our case, the trust level is determined by the reasoning on the presence/absence of trust dependencies in the early requirements model. The trust level value denotes the level of trust between the truster and the trustee on the fulfilling of the BP. The determined trust level of service providers might be employed when there is a possibility to choose one BP from the several alternatives suggested by different providers. The advantage of the "0/1 trust level

approach" is to have the possibility of choosing the best alternative, i.e., the right partner to work with, in case of distrust to another one.

2.3. Goal-oriented requirement engineering for services

The emergence of (Web) services has called for new methodologies for designing systems. In Papazoglou and Yang (2002), basic principles of Web services and business processes design are presented. The early work on this issue already offers promising insights, though it does not distinguish logical BPs and their implementation, when our approach produces executable secure BPs with SLAs.

A promising requirement engineering approach to service based systems, which we adopt in the current treatment, is that based on goal-orientation. Goal-Oriented requirement engineering has been proposed and adopted before the introduction and later popularity of Web services, see for instance the overview in Lamsweerde (2001) and the reported experiences in Rolland et al. (1999). The application to Service-Oriented Architectures is more recent and mainly two approaches have been proposed and worth to be mentioned: intentional one presented by the work by Rolland et al. and a set of approaches that build on the Tropos goal-oriented methodology.

Rolland et al. (2010) introduce the notion of *intentional services* and propose a methodology that allows to derive operational services from intentional ones and then, to map those onto software executable services. The methodology goes from the abstract design level, to composition and then implementation, providing a solid complete framework. Though, the dealing of non-functional aspects over business processes is not dealt with.

The *Tropos methodology* (Castro et al., 2002; Bresciani et al., 2004) is a requirements engineering methodology that supports all analysis and design activities in the software development process, from application domain analysis to system implementation. Lau and Mylopoulos (2004) propose a design methodology for Web services adapted from the Tropos project. The work is based on the use of goals to determine the space of alternative solutions to satisfy them. The key point is that the solutions are represented as Web services. The generated Web service design is expected to accommodate as many of those solutions as possible, thus making the design usable by a broader class of applications. On the negative side, Tropos is not tailored specifically to Web service design. Therefore the methodology does not address the issue of integration neither of Web Service Business Process Language in order to specify actual behavior of participants in a business interaction nor WS-Agreement Language to specify SLAs of the services. Kazhamiakin et al. (2004) propose a methodology for business requirements modeling that uses the Tropos framework to capture the strategic goals of the enterprise. The proposed methodology allows the creation of concrete BPs expressed as BPEL4WS descriptions. The concrete BPs are elicited from the description of BP notions with Tropos concepts extended with a formal annotation called Formal Tropos (Fuxman et al., 2004). On the other hand, our work aims not only to obtain BPs from an early requirements, but also to enrich them with SLAs. Penserini et al. (2006) address the issue of refining the Tropos methodology and tailoring it to the design of Web services. The Tropos design process is extended to support a revised notion of capability that explicitly correlates actor plans with stakeholders needs and environmental constraints. The agent capability is considered as a service. Furthermore, the authors sketch how Tropos design-time models can support service discovery and composition by relating stakeholder goals to sets of services available. Even if the idea is feasible, the work is in an early stage and there is a need for more precise mapping of agent capability that is considered as a service. Furthermore, there is no secure BPs design support.

The secure Tropos framework (Giorgini et al., 2006; Massacci et al., 2007) allows to derive the secure requirements towards the secure BPs. Secure Tropos extends the Tropos methodology by adding security concerns during the development process to introduce concepts such as ownership, trust, and delegation within a requirements modeling framework and shows how security and trust requirements can be derived and analyzed. The main advantages in using the Tropos methodology are that it allows to capture not only the *what* or the *how*, but also *why* a piece of software is developed. This, in turn, allows for a more refined analysis of the system dependencies and, in particular, for a much better and uniform treatment not only of the system functional requirements, but also of its non-functional requirements.

Lapouchnian et al. (2007) propose a requirements-driven approach for business process design. Requirements goal models are used to capture business goals and alternative process configuration. Quality attributes such as customer satisfaction serve as the selection criteria for choosing among BPs alternatives induced by the goal models. Executable BPs are generated in a semi-automatic way starting from goal models. The approach does not focus neither on secure business processes nor SLA building for generated business processes.

Naturally, also non goal-oriented methodology have been proposed for the design of service-based business processes. A methodological approach for deriving the software functionality from an organizational model is presented in de la Vara et al. (2008). The authors model an organization by means of BPMN and use the goal/strategy Map approach. The work allows for organization analysis and system goals understanding in a participative way with customers. The approach does not focus on non-functional properties of BPs.

Distante et al. (2007) analyze and compare Web applications design methodologies with regards to their support for modeling business processes. Further, a comprehensive design model for integrated BPs in Web applications is proposed. The model is based on UWAT+, an extension of the ubiquitous Web applications design model called UWA. The proposed model satisfies plenty of requirements, while it does not work with non-functional properties and SLAs.

3. Collaborative procurement in the Sekhukhune Rural Living Lab

In the area of procurement, the quality of the execution of a process is crucial. Consider the Collaborative Procurement&Logistics use case in the Sekhukhune Rural Living Lab (RLL). The scenario is provided courtesy of SAP[2] and is a working scenario of the IST-FP7-IP-C@R research project.[3]

The Sekhukhune District, located in the Limpopo Province, is a prime example of an underdeveloped rural area in South Africa. The region suffers from severe infrastructural bottlenecks in almost all sectors (basic services, health, education, transport and communication to name just a few). The C@R project is developing mechanisms to support informal micro enterprises daily business needs. The goal is to make them more efficient and effective, linking them up with the established formal economy and thus supporting economic development in the Sekhukhune RLL. Both from a business and from a technical perspective, the situation calls for innovative user-centric approaches going beyond the established norm. Several use cases are going to be implemented in the Sekhukhune RLL: Collaborative Procurement&Logistics,

[2] http://www.sap.com.
[3] http://www.c-rural.eu.

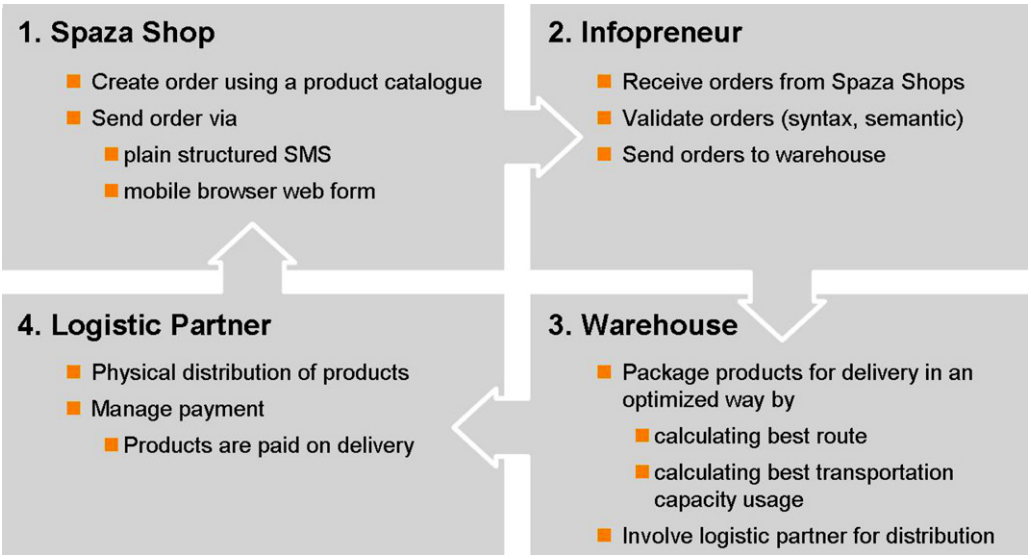

Fig. 1. Collaborative Procurement&Logistics use case in the Sekhukhune RLL.

Collaborative Stock Management&e-Commerce and Collaborative Knowledge Sharing.

A more detailed description of the Collaborative Procurement&Logistics use case with all stakeholders, their goals and relations is depicted in Fig. 1. Currently the described features "Mobile browser Web form" and the "Package products for delivery in an optimized way" are still in the prototype status while everything else is already successfully piloted in the Sekhukhune RLL for a period of 9 month.

The Collaborative Procurement&Logistics use case contains four main actors: `Spaza Shop` (small retail enterprises operating from a residential home or small store, selling basic goods to the community), `Infopreneur`™ (Rensburg et al., 2008) (self-sustainable, service entrepreneur providing information and data related services in rural areas), `Warehouse` and `Logistic Partner`. The aim of the BP is to place an order and to get the products delivered. The Spaza Shops are organized in a virtual cooperative (Merz, 2010) to enjoy the benefits of better bargaining power, bulk orders and price reductions.

Depending on the network coverage/cost and on ownership of a Java enabled phone, a Spaza Shop could either use a simple paper-based product catalogue in conjunction with a structured order SMS or an on-line product catalogue accessed by a mobile phone Web browser to create an order. With a low end mobile phone, one can send a structured SMS to a dedicated number. The procurement system fetches all incoming orders for further processing (syntax check, authorization). When the Spaza Shop owner owns a Java enabled phone, the mobile Web browser is used to place the order via a GPRS post request triggered by a Web form. Because of infrastructural (low, erratic and expensive bandwidth, erratic power supply, low end devices, limited remote support) and cultural (illiteracy, a variety of local languages, low ICT knowledge and different device usage behaviors) impediments (Doerflinger et al., 2009) a "always on-line" application realized as a desktop application running on a desktop PCs to send the order is not applicable in the Sekhukhune rural area. Mobile phones form the only and broadly used communication channel in this area, which is common of rural areas of developing countries.

After the order has been placed and accepted as a valid one in the procurement system, the Infopreneur™ uses an offline capable application to download and process the new orders. The Infopreneur's™ task is to check if the orders are semantically correct. It is a kind of an intermediate or sales agent where all incoming orders are processed, bundled and forwarded to the warehouse. It acts as the locally responsible person for the connection between Spaza Shops and Warehouse.

At the Warehouse the incoming orders are bundled and prepared for transportation by the warehouseman. The packaging of the ordered products is made on the basis of optimization parameters. To offer the products to the Spaza Shop for the best possible price, it is necessary to optimize the transportation with respect to the optimal route and transportation capacity use for the trucks. The Warehouse then involves the logistic partner and provides the information about route and capacity planning.

The Logistic Partner is responsible for the physical distribution of the products to the Spaza Shops. The payment is realized as "pay on delivery." That means that the Logistic Partner collects the money from the Spaza Shop at the time he delivers the products. As several logistic partners can be involved, some partners might be more trustworthy and reliable than others. The warehouse needs to take into account which logistic partner is the most appropriate for a specific delivery. Trust in the delivery process is a key factor because the logistic partner is also responsible for the payment.

From the business point of view, the process is quite common. Though formal modeling and having tools for its management in a robust manner are a challenge. Furthermore, in the area of procurement, both quality of execution and security of a process are crucial.

4. The BP&SLA methodology

Judging the appropriate SLA to sign after having defined the business objectives is far from being a straightforward task. With the Business Processes with Service Level Agreements (BP&SLA) methodology, we provide means to go from a high-level analysis of the business requirements all the way to the definition of the processes to be executed and the SLAs to be signed in order to guarantee certain quality of service. The methodology consists of four main phases which are, referring to Fig. 2, (1) early requirements engineering, (2) business process hypergraph derivation, (3) hierarchy of business processes derivation, and (4) constraint reasoning for Service Level Agreements derivation.

During the first phase, the end user or domain expert provides informal requirements that form the seed for developing formal processes. These early requirements are manually formalized following the Secure Tropos methodology, an extension of the well

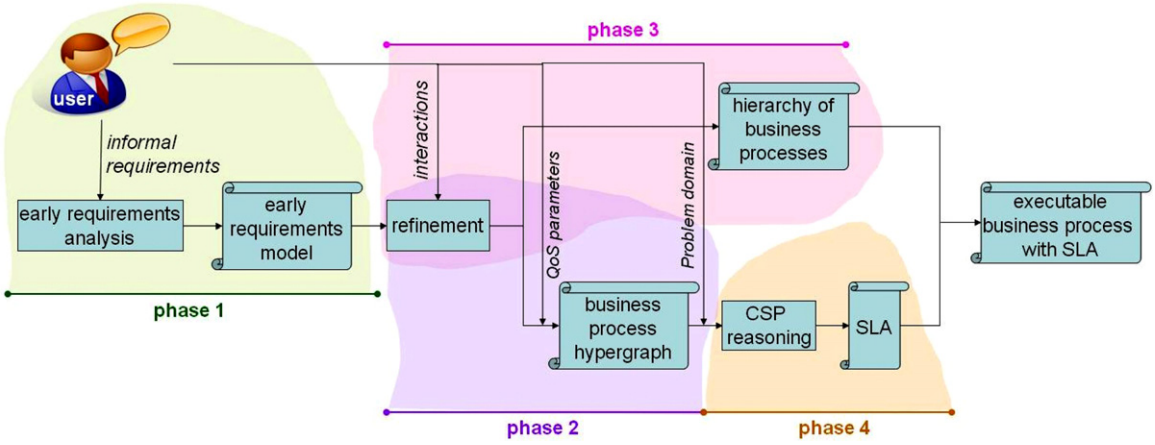

Fig. 2. The BP&SLA Methodology.

established Tropos software engineering methodology (Bresciani et al., 2004). The output of this phase is an early requirement model. The model is far from being an executable entity, but rather it is a conceptual description of the actors involved in the business, their goals and their trust and security relations. To transform the model into something executable, in the second and third phase, one navigates automatically the model. The user is possibly asked to disambiguate choices with more than one interpretation. The results of the refinement of the early requirements are an intermediate model on which to perform qualities of service inference: the BP hypergraph, and a hierarchy of BPs. Such processing occurs in Phases 2 and 3. The BP hypergraph is further analyzed to build a constraint problem which represents the relationships among the various elements of the processes regarding quality of service and security properties of the processes. By reasoning with these constraints it is possible to derive the appropriate SLAs to be signed to guarantee a certain quality of service (Phase 4). This phase is entirely automated. The final output of the methodology is a hierarchy of BPs ready for execution together with SLAs fulfilling a specific quality of service. Let us consider next each of these phases individually. We consider the collaborative procurement&logistics scenario in the Sekhukhune Rural Living Lab presented in Section 3 as a running example.

4.1. Phase 1. Early requirements engineering

Early requirements engineering aims at analyzing the organizational context within which a system will eventually operate. During an early requirements analysis the domain actors and their dependencies on other actors for goals to be fulfilled are identified. For early requirements model elicitation in the context of security, one needs to reason about trust relationships and delegation of authority.

We employ the Secure Tropos modeling framework (Giorgini et al., 2006; Massacci et al., 2007) to derive and analyze both functional dependencies and security and trust requirements. For the acquisition of the early requirements model we employ several modeling activities. Actor modeling to identify the principal stakeholders (actors) and their objectives (goals). Each goal might be refined by AND/OR goal decomposition that AND/OR decomposes a root goal into sub-goals. It might happen that an actor does not have the capabilities to achieve his own objectives by himself. In this case that actor has to delegate the objectives to other actors leading to their achievement outside the control of the delegator. Secure Tropos supports two types of delegations. *Delegation of execution* occurs when one actor delegates to another one the responsibility to execute a service. It is also known as 'at-least delegation.' *Delegation of permission* models the transfer of entitlements from an actor to another one, also known as 'at-most delegation.' We use functional dependency modeling to identify actors depending on other actors for obtaining services, and actors which are able to provide services. Permission delegation modeling is used to identify actors delegating to other ones the permission on services. Secure Tropos supports two types of trust dependencies. *Trust of execution* occurs when one actor trusts that another one will at least fulfill a service. This is also known as 'at-least trust.' While the meaning of *trust of permission* is that an actor trusts that another one will at most fulfill a service, but will not overstep it. This is also known as 'at-most trust.' Trust modeling aims at identifying actors trusting other actors for services, and actors which own the services.

Performance-based trust model

We propose a KPI based trustworthiness model[4] taking into account the business objectives described previously, and assigning automatically trust level values (0 or 1).

The traditional trustworthiness models deployed in famous online shops such as Amazon or eBay are relying on a subjective rating system in which users estimate the "quality" of the transaction over a numerical scale. Knowing that nobody is able to formalize and explain the difference between two successive values like a transaction rewarded at 9/10 and another one 10/10, we can not really estimate the correctness and the objectivity of the trust and reputation value.

We propose a less subjective trust model taking into account the performance of each business partner according to their business objectives or to a business agreement like SLA. For example, if a business partner does not satisfy a target in the SLA, he will be penalized. Each trustee entity chooses the business objectives that must be satisfied by the partners to trust. These objectives must be measurable like a set of performance indicators, e.g., price, time, packaging, payments conditions, quality of service.

After each interaction between two business partners, the trustee gets these quantifiable values and compares it to the objectives in order to obtain trust indicator values. These indicator values are then aggregated and normalized in order to obtain a unified trust level value. The model is shown in Fig. 3 and it is composed of three complementary layers:

Performance Indicator Values are collected and calculated after each execution of the entire life-cycle in the Collaborative Procure-

[4] This Performance-based trust model was elaborated in the context of IST-FP7-IP-TAS3 European Project. http://www.tas3.eu/.

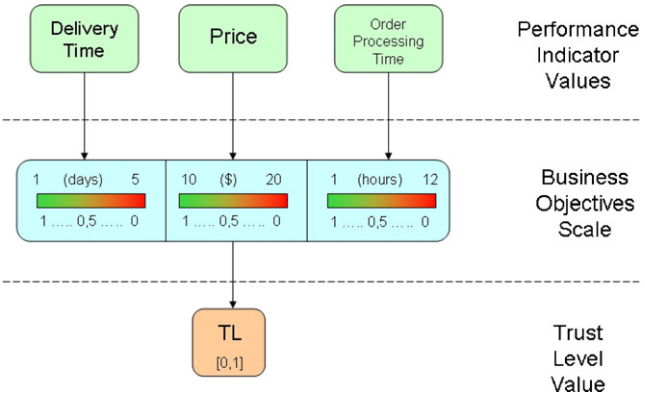

Fig. 3. Performance-based trust model.

ment & Logistics use case, then compared to the business objective scale.

Business Objectives Scale are fixed by the trustee according to the performance indicators related to their business objectives. An interval of values (min and max) must be chosen for every performance indicator in order to normalize the measured value on a [0,1] scale. The [0,1] normalization rule is as follow:

$$\begin{cases} 1 & \text{if} \quad K_i > K_{\max} \\ \dfrac{K_i - K_{\min}}{K_{\max} - K_{\min}} & \\ 0 & \text{if} \quad K_i < K_{\min} \end{cases}$$

where K_i is the measured performance indicator value, $K_{\min}$ and $K_{\max}$ are the maximum and minimum values declared in the business objectives scale.

Trust Level Value is the aggregation of all the normalized performance indicators plus eventually some external values such as the recommendation from other trusted entities.

In summary, the performance based trust model offers the possibility to quantify the trustworthiness values according to business objectives and SLAs and permit to any BP component to determine which business partner is more trustable according to an objective estimation. Usually in traditional recommendation systems, the trustee relies on a binary recommendation value. In our performance-based trust model the trustee can evaluate the weight of a recommendation by accessing to the business objective scale of the recommender.

4.2. Phase 2. Business process hypergraph derivation

The second phase of the BP&SLA methodology is devoted to creating an intermediate structure to reason about the BPs and their qualities. This intermediate structure is an hypergraph, which we define as follows.

Definition 1. A **business process hypergraph** $\mathcal{B}$ is a pair $\langle B, H\rangle$ where B is a set of business processes and H is a set of hyperarcs. A *hyperarc* is an ordered pair $\langle N, t\rangle$ from an arbitrary nonempty set $N \subseteq B$ (source set) to a single node $t \in N$ (target node). Each hyperarc is associated with a vector of aggregation functions $\varphi = [\varphi_1\langle N, t\rangle, \ldots, \varphi_n\langle N, t\rangle]$ which calculate value of a target node taking as arguments source nodes, with the structural activity associated, for a particular QoS parameter.

Consider QoS parameters as maximal execution time (Max ET), availability (Av) and maximal time to recover after an attack (Max TR) for sequential, parallel and choice structural activities are as follows:

Activity	Max ET	Av	Max TR
Sequence	$\varphi = \sum_{i=1}^{n} p_i$	$\varphi = \prod_{i=1}^{n} p_i$	$\varphi = \sum_{i=1}^{n} p_i$
Parallel	$\varphi = \sum_{i=1}^{n} p_i$	$\varphi = \prod_{i=1}^{n} p_i$	$\varphi = \sum_{i=1}^{n} p_i$
Choice	$\varphi = \max(p_1, \ldots, p_n)$	$\varphi = \min(p_1, \ldots, p_n)$	$\varphi = \max(p_1, \ldots, p_n)$

Other examples of functions are illustrated in Jaeger et al. (2005).

The BP hypergraph is obtained by navigating the early requirement model and refining it, possibly with user interaction. This is performed algorithmically according to the procedure presented in Appendix A. The algorithm takes the early requirements model, the actor with its goal and the vector of QoS parameters as an input. Each node of the BP hypergraph is a BP that corresponds to a goal in the early requirements model. As we consider the goals to be operational, each hyperarc in the BP hypergraph corresponds to the goal refinement or delegation dependency in the early requirements model. The concept of AND goal decomposition is refined as sequential or parallel BP composition in the BP hypergraph. The concept of OR goal decomposition in the early requirements model is refined as branching statement in the BP hypergraph. The aggregation function for sequential, parallel or choice aggregation of QoS parameters are applied. In case of a specific design choice, the nodes corresponding to the BPs are connected by different hyperarcs with the target node. The design choice structural activity appears in case of the presence of different alternatives for the same BP, e.g., the same BP might be delegated to different partners that have different SLA offers. The refinement of the concept of AND/OR goal decomposition from the early requirements model can not be completely automated, but only supported and involves user interaction.

Each node in the BP hypergraph is assigned with a vector of QoS parameters and a Trust Level value (TL). The values of the QoS parameters correspond to the QoS that can be achieved by the BP. The trust level value denotes the level of trust between the truster and the trustee on the fulfilling of the BP. In Frankova and Yautsiukhin (2007), we propose a methodology that identifies the concrete BP providing the highest quality of service and protection among all possible design alternatives. The idea is to take into account the level of trust of service providers and adjusts the expected quality value correspondingly. In spite of the fact that the approach of using the notion of trust as weighting factor is promising, the authors do not clarify how the trust values are decided. Instead in our approach the trust level is determined from the reasoning on the presence/absence of trust dependencies in the early requirements model. Then, when the BP with SLA is in place, we apply the proposed performance-based trust model in order to determine the partner to work with when there is a possibility to choose one BP from the several alternatives suggested by different providers.

4.3. Phase 3. Hierarchy of business processes derivation

The third phase of the BP&SLA methodology is dedicated to hierarchy of BPs construction. We build the hierarchy of BPs with the aim to use it for obtaining a set of executable secure BPs. These are created following the Secure BPEL specifications (Frankova et al., 2007a; Sèguran et al., 2008). Secure BPEL is a dialect of WS-BPEL for the functional parts and abstracts away low level implementation details from WS-Security and WS-Federation specifications. Secure BPEL allows to describe delegation relations (both of execution and of permission) and trust relations (both on execution and on permission) among all the partners that execute sub-BPs in the context of the global BP. In the hierarchy of BPs, each delegated BP is labeled with an SLA which will be derived in Phase 4. The hierarchy of BPs, as well as the BP hypergraph, is derived by refining the early requirements model. As we build the hierarchy to obtain executable BPs with SLAs, we must clearly determine (1) the BPs,

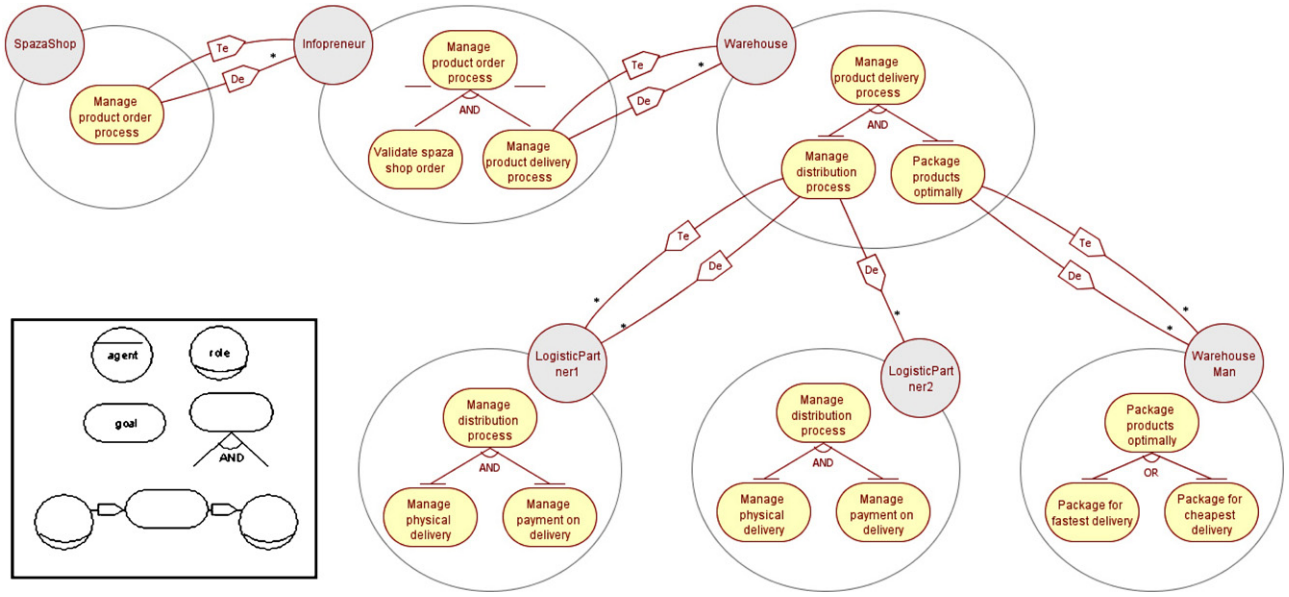

Fig. 4. Early requirements model for Collaborative Procurement&Logistics use case.

(2) which partner proceeds which BP, and (3) delegation and trust dependencies among the involved partners.

The main idea is that analogously to the BP hypergraph construction, we consider the level of goals in the early requirements model to be the level of BPs in the hierarchy of BPs. Furthermore, the BP(s) proceeded by one actor are grouped and marked with the actor. Each partner has to know which BP to proceed. For the detailed presentation of the algorithm we refer to Frankova et al. (2007b).

In this work, we adopt only the Secure Tropos delegation of execution dependencies, but not the delegation of permission ones to label with SLAs only the BPs that are delegated to be executed. We consider the fact that one needs to sign an SLA with the partner only in case of transfer of responsibilities to the partner, i.e., the BP is delegated to the partner and the partner processes it. While if there is only a fact of transfer of entitlements, i.e., the BP is delegated to the partner and the partner has permissions to processes the BP, but do not actually does it, there are no reasons for an SLA signing. Further, we employ both at-least and at-most trust and delegation notions to implement the relations between the actors in the hierarchical structure of BPs.

4.4. Phase 4. Constraint reasoning for SLAs derivation

In the last phase of the BP&SLA methodology, SLAs for BPs are derived by reasoning on the BP hypergraph. The reasoning technique we employ is constraint programming. The key idea is to state the relationships among the qualities of processes and their activities as a set of constraints. Formally, the Constraint Satisfaction Problem (CSP) is defined as follows (Tsang, 1995):

- a set of variables $\{x_1, \ldots, x_n\}$,
- for each variable x_i a finite set D_i (its domain) of possible values,
- a set of constraints, i.e., relations or expressions, restricting the values that the variables can simultaneously take.

A solution to CSP is an assignment to the set of variables such that all its constraints are satisfied. One may want to find an optimal solution, if some objective function is given over CSP variables.

We build a constraint systems by recursively navigating the BP hierarchy and hypergraphs. The algorithm is presented in Appendix B. The algorithm takes the BP hypergraph, the node to start with, and the problem domain as an input, and it builds a constraint expression for every level of the hypergraph. Intuitively, the expression represents the quality of service for that level. For each level a new fresh variable is added and its range is restricted to the domain of the quality of services. Depending on what kind of children are available for that level different kind of expressions are built. If the children are connected with AND, the expression is built as an aggregation of the variables representing the children nodes. In the case of choice, there are different expressions for each child and an additional expression represents the fact that only one child will contribute to the execution (the sum of x_i). Once the constraint expressions are built, the algorithm proceeds recursively on all children. If the node is a leaf node, then one simply adds a variable for that node and a constraint on the domain of the variable.

Once the constraint system is in place, one can perform constraint propagation to find the solution space for acceptable qualities of services. If then one desires to have SLAs to attach to the BPs, it is simply a matter of performing a labeling of the solution space and obtaining satisfying values for the qualities of services. We remark that such a solution might not exists. In this case, the result of the methodology will be a set of processes, but with no quality guarantees.

5. Applying the methodology

To better understand the methodology just presented, consider the collaborative procurement use case in the Sekhukhne Rural Living Lab presented in Section 3 and apply the proposed methodology.

5.1. Phase 1

The early requirement model for the collaborative procurement and logistics use case is depicted in Fig. 4.

The early requirement model presents the principle entities involved, (1) actors depicted as circles and (2) interests, i.e., goals, presented as ovals. The `SpazaShop` actor has the goal to `Manage product order process`. This goal is delegated to the `Infopreneur`™ actor. The delegation of execution is depicted with two lines connected by the delegation of execution (De) mark.

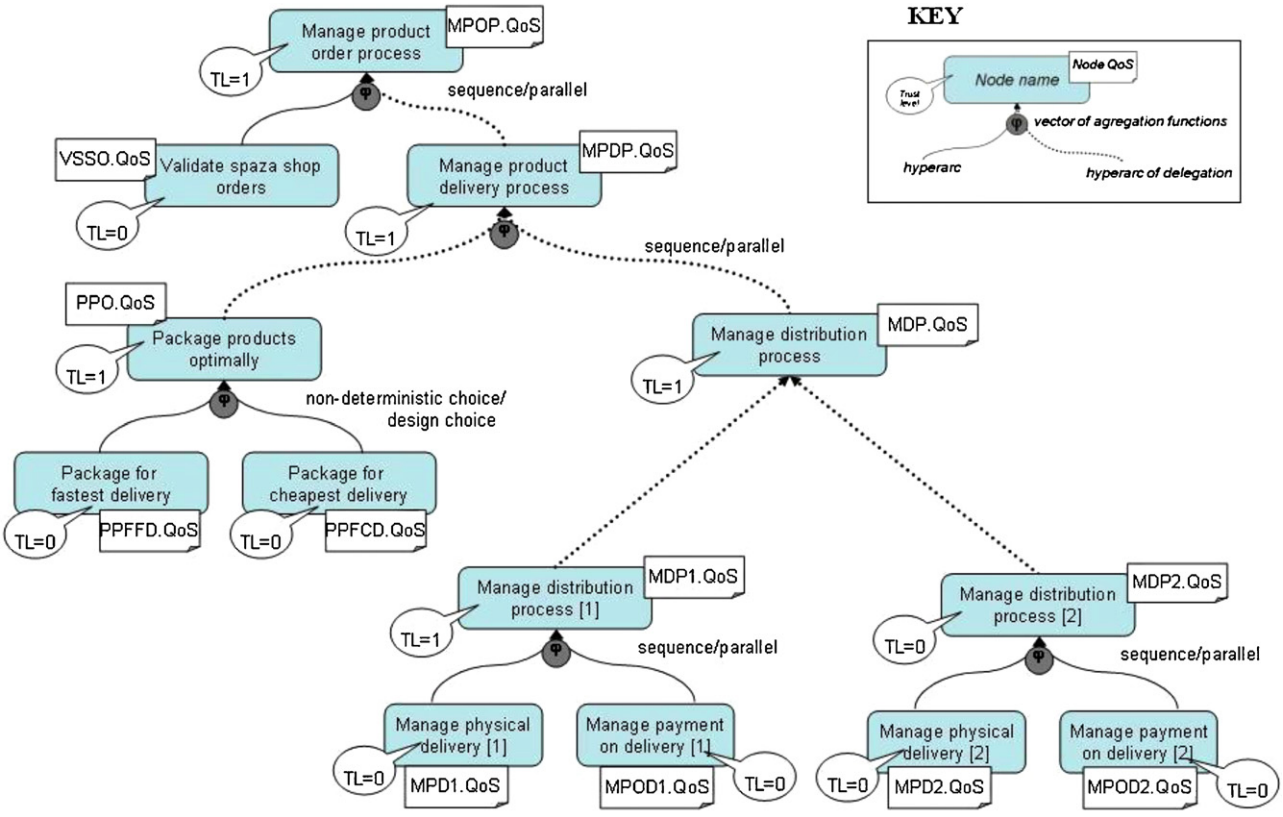

Fig. 5. Business Process Hypergraph for Collaborative Procurement&Logistics use case.

The `SpazaShop` actor trusts the `Infopreneur`^{TM} actor on execution of the goal. The trust on execution is depicted with two lines connected by the trust execution (Te) mark. In order to fulfill the goal the `Infopreneur`^{TM} actor refines it by an AND decomposition, depicted with a goal refinement symbol marked with AND, into goals to `Validate Spaza shop orders` and to `Manage product delivery process`. The `Infopreneur`^{TM} actor delegates the latter goal to the `Warehouse` actor. Again the delegation on execution and trust on execution is used. The `Warehouse` actor refines the `Manage product delivery process goal` into `Manage distribution process` and `Package products optimally`. The latter goal is delegated to the `WarehouseMan` actor. And delegation of execution and trust on execution are used. The `WarehouseMan` actor refines the goal by an OR decomposition, depicted with a goal refinement symbol marked with OR, into the `Package for fastest delivery` and `Package for cheapest delivery goal`.

The `Manage distribution process goal` at the `Warehouse` actor is delegated to the `LogisticPartner1` and `LogisticPartner2` actors. While the `Warehouse` actor trusts on execution to the `LogisticPartner1`, there is no trust relation between the actor and the `LogisticPartner2`.

The `LogisticPartner1` and `LogisticPartner2` actors refine the `Manage distribution process` goal into the `Manage physical delivery` and `Manage payment on delivery` goals.

As for the trust model schematized in Fig. 3, the delivery time can be chosen by the Spaza shop as a performance indicator. According to Spaza's business objectives the delivery delay must be comprised between $K_{min} = 1$ day and $K_{max} = 5$ days. Using this scale we normalize the delivery time values in order to be fitted to a [0,1] scale. For example, if the delivery time is $K_i = 3$ days, in the middle of the boundaries (not so good $K_{max} = 1$ day and not so bad $K_{max} = 5$ days) the trust value will be 0.5.

5.2. Phase 2

The hypergraph of Phase 2 corresponding to the case depicted in Fig. 4 is shown in Fig. 5.

The global goal of the Business Process Hypergraph depicted in Fig. 5 is `Manage product order process`. The BPs `Validate Spaza shop orders` and `Manage product delivery process` are delegated BPs that contribute to the satisfaction of this global goal. They are connected to it by a dashed hyperarc, starting from the delegated BP pointing to the target BP. Each node (BP) is assigned a vector of QoS parameters and trust level values while each hyperarc is assigned a vector of aggregation functions φ. The aggregation function takes into account the structural activity and the QoS parameter, i.e., sequence flow or parallel flow in this case, associated to the delegated BPs `Validate Spaza shop orders` and `Manage product delivery process`.

Both BPs, `Package products optimally` and `Manage distribution process`, are delegated BPs of the `Manage product delivery process` BP they are contributing to and thus are connected with a dashed hyperarc. Again a vector of aggregation functions φ is assigned to the hyperarc taking into account the structural activity associated to the two delegated BPs. In this case, a sequential/parallel flow is assigned to the hyperarc.

The BP `Package products optimally` has two contributing delegated BPs, `Package for fastest delivery` and `Package for cheapest delivery`. In this case the vector of aggregation functions assigned to the hyperarc represents a non-deterministic choice or design choice (in which case there would be two different hyperarcs).

The delegated BPs `Manage distribution process (1)` and `Manage distribution process (2)` are connected with two

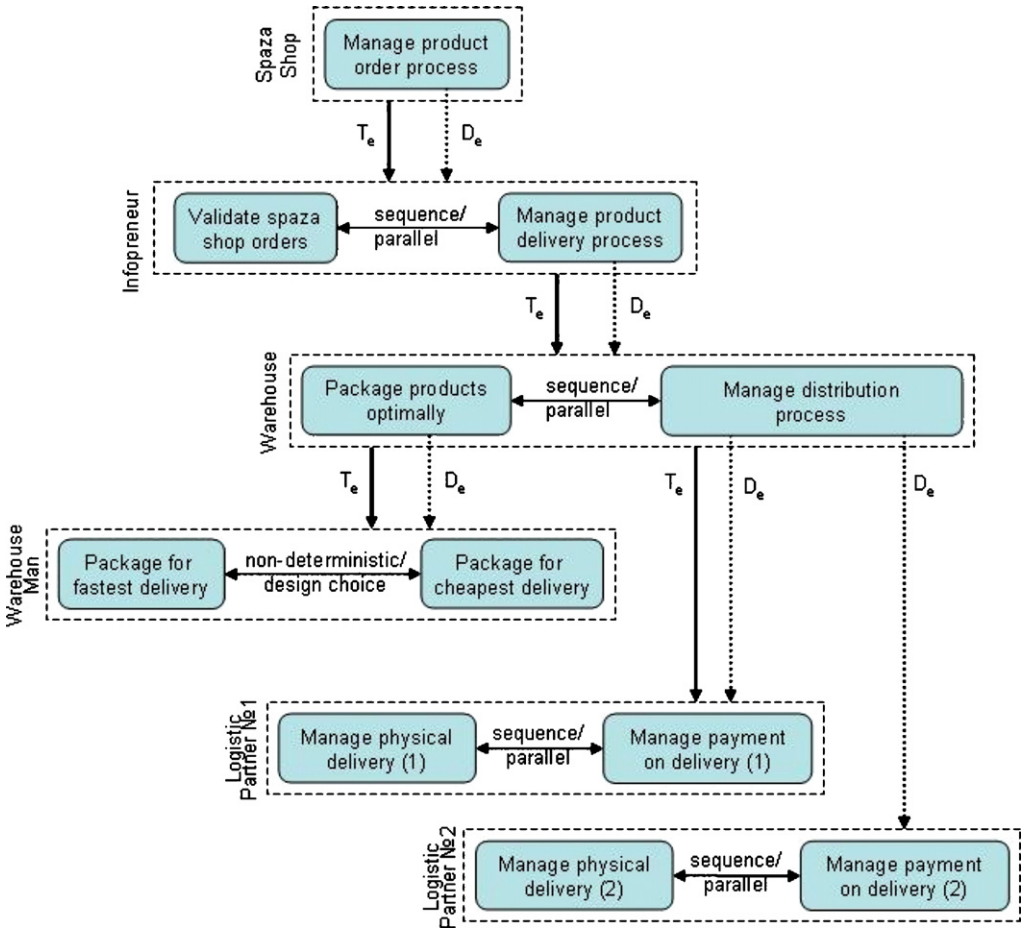

Fig. 6. Hierarchy of Business Processes.

different hyperarcs to the `Manage distribution process` BP they are contributing to using dashed hyperarcs.

The nodes `Manage physical delivery (1)` and `Manage payment on delivery (1)` are delegated BPs of the `Manage distribution process (1)` BP. And the nodes `Manage physical delivery (2)` and `Manage payment on delivery (2)` are delegated BPs of the `Manage distribution process (2)` BP. In both cases a vector of aggregation functions φ is assigned to the hyperarc that takes into account the structural activity associated to the `Manage physical delivery (1)/(2)` and `Manage payment on delivery (1)/(2)` BPs and the QoS parameter. In this case a sequential/parallel flow is assigned to the hyperarc.

The individual trust level (TL) is determined from the reasoning on the presence/absence of trust dependencies in the early requirements model. After the BP with SLA is in place, we apply the proposed performance-based trust model in order to determine the partner to work with when there is a possibility to choose one BP from the several alternatives suggested by different providers.

5.3. Phase 3

The result of Phase 3, that is the hierarchy of BPs corresponding to the early requirement model for the manage product order process case study is shown in Fig. 6. Each goal is associated with a BP, represented by a rounded-corner rectangle in the hierarchy. Dashed rectangles are used in order to represent the actors that proceed the BPs. In our case these actors are the `SpazaShop`, the `Infopreneur`^{TM}, the `Warehouse`, the `WarehouseMan`, the `LogisticPartner1`, and the `LogisticPartner2`.

The dependencies among actors, i.e., delegation and trust, are represented as dashed and solid connectors with corresponding marks. The `SpazaShop` actor delegates execution of the `Manage product order process` BP to the `Infopreneur`^{TM} actor. The `SpazaShop` actor trusts the `Infopreneur`^{TM} actor to fulfill the `Manage product order process` BP. These facts are depicted by the proper connectors.

The relation among BPs proceeded by the `Infopreneur`^{TM} actor is defined by the structural activity associated to the `Validate Spaza shop orders` and `Manage product delivery process` BPs. The `Infopreneur`^{TM} actor delegates execution of the `Manage product delivery process` BP to the `Warehouse` actor. While the `Infopreneur`^{TM} actor proceeds the `Validate Spaza shop orders` BP on his own. The `Infopreneur`^{TM} actor trusts the `Warehouse` actor to fulfill the `Manage product delivery process` BP. The relation among BPs proceeded by the `Warehouse` actor is defined by the structural activity associated to the `Package products optimally` and `Manage distribution process` BPs.

The `Warehouse` actor delegates execution of the `Package products optimally` BP to the `WarehouseMan` actor and trusts that the `WarehouseMan` will fulfill the `Package products optimally` BP. The relation among BPs proceeded by the `WarehouseMan` actor is defined by the structural activity associated to the `Package for fastest delivery` and `Package for cheapest delivery` BPs. The structural activities are defined by corresponding goal decomposition types in the early requirements model.

The `Warehouse` actor delegates execution of the `Manage distribution process` BP to the `LogisticPartner1` and `LogisticPartner2` actors. There are no trust dependencies

MPOP=MPOP.ET+sum(VSSO.ET,MPDP)
MPDP=MPDP.ET+sum(PPO,MDP)
PPO=PPO.ET+max(PPFFD.ET,PPFCD.ET)
MDP=MDP.ET+MDP1*x_1+MDP2*x_2 where $x_i \in 0,1$ and $\sum x_i = 1$.
MPDP1=MPDP1.ET+sum(MPD1.ET,MPOD1.ET)
MPDP2=MPDP2.ET+sum(MPD2.ET,MPOD2.ET)

Fig. 7. Quality constraint expressions.

between the Warehouse actor and the LogisticPartner1 actor on the Manage distribution process BP. While trust on execution connects the trusted BP Manage distribution process with the trustee LogisticPartner2. The relations among BPs proceeded by the LogisticPartner1 and LogisticPartner2 actors are defined by the structural activities associated to the Manage physical delivery (1)/ (2) and the Manage payment on delivery (1)/ (2) BPs.

5.4. Phase 4

Finally, we show the generation of SLAs based on given quality of service requirements for the execution of the BP using the Collaborative Procurement&Logistics case study. The example is based on the real data coming from an actual case study. In order to obtain the quality constraint expressions, we need to be given the domain over which the quality of services range, e.g., integers for costs or real numbers for response time. In the case of the proposed methodology, the QoS Domain is a vector of QoS with corresponding possible values for the parameters. The example of the QoS Domain we consider is the following vector: [Execution Time (ET) ∈ N, Availability (Av) ∈ N, Time to Recover after an attack (TR) ∈ N].

Several examples of the aggregation functions for such QoS parameters as maximal execution time (Max ET), availability (Av) and maximal time to recover after an attack (Max TR) for sequential, parallel and choice structural activities are presented in Jaeger et al. (2005).

Fig. 7 presents the quality constraint expressions obtained for the maximal execution time parameter navigating the BP hypergraph from Fig. 5 following the algorithm. Where MPOP stands for Manage Product Order Process, VSSO for Validate Spasa Shop Orders, MPDP for Manage Product Delivery Process, PPO for Package Products Optimally, MDP to Manage Distribution Process, PPFFD for Package for Fastest Delivery, PPFCD for Package for Cheapest Delivery, MDP1 and MDP2 for Manage Distribution Process(1) and Manage Distribution Process(2), MPD1 and MPD2 for Manage Physical Delivery(1) and Manage Physical Delivery(2), MPOD1 and MPOD2 for Manage Payment on Delivery(1) and Manage Payment on Delivery(2) BPs. The expressions for such parameters as availability and maximal time to recover after an attack can be found in Frankova et al. (2007b).

In the running example the constraint propagation for maximal execution time QoS property for the super-process in order to achieve execution time less then 60 s is performed and we get the following satisfying values for the qualities of services: MPD1.ET =10s, MPOD1.ET =5s, MPD2.ET =5s, MPOD2.ET =3s, MDP1.ET =6s, MDP2.ET =2s, MDP.ET =12s, PPFFD.ET =3s, PPFCD.ET =2s, PPO =10s, MPDP =4s, VSSO =5s, MPOP =4s.

The SLAs for the delegated BPs are presented in Fig. 8 where

SLA(MPOP)=MPOP=MPOP.ET+sum(VSSO.ET,MPDP)=4s+sum(5s,50s)=59s
SLA(MPDP)=MPDP.ET+sum(PPO,MDP)=4s+sum(13s,33s)=50s
SLA(PPO)=PPO.ET+max(PPFFD.ET,PPFCD.ET)=10s+max(3s,2s)=13s
SLA(MDP)=MDP.ET+MDP1*x_1+MDP2*x_2=12s+21s*1+10s*0=33s

Fig. 8. SLAs for the delegated BPs.

MDP1=MDP1.ET+sum(MPD1.ET,MPOD1.ET)=6s+sum(10s,5s)=21s
MDP2=MDP2.ET+sum(MPD2.ET,MPOD2.ET)=2s+sum(5s+3s)=10s

Note that while choosing the BP among two alternatives Manage Distribution Process(1) and Manage Distribution Process(2) we rely on the trust levels of the providers of the services. As the trust level of the first provider is 1 and the one of the second provider is 0, we choose the first option. Then, when the BP with SLA is in place, we apply the proposed performance-based trust model in order to determine the partner to work with when there is a possibility to choose one BP from the several alternatives suggested by different providers.

Finally, the obtained SLA is described using WS-Agreement language (Andrieux et al., 2007). We remark that, using the framework proposed in Frankova et al. (2006), one could monitor the agreement with the option to anticipate violations and renegotiate at runtime the guarantees.

6. Prototype

The constraint algorithm for SLAs (Appendix B) is implemented in the constraint satisfaction framework ECLiPSe[5] using the IC Hybrid Domain Solver.[6] The basic algorithm is a typical tree walking algorithm that can be applied to any sub-tree of a BP hypergraph (BPH). Constraints are applied from the leafs up, to make the algorithm fail early if a low-level constraint cannot be satisfied. All constraints over multiple child nodes are expressed as a user-defined predicate.

The core algorithm is implemented in two small predicates described next.

```
cspec(BPH, QoSDomain) :-
  qos(BPH, QoSValue),
  %constrain the QoS-Variable present in the current node
  %to the solution domain
  QoSValue #:: QoSDomain,
  %apply the cspec_algorithm on all children
  children(BPH, Children),
  cspec_children(Children, QoSDomain),
  %apply the function phi given for the current node
  apply_fun(BPH).

cspec_children([],_). cspec_children([Head—Tail],QoSDomain):-
  cspec(Head, QoSDomain),
  cspec_children(Tail, QoSDomain).
```

The algorithm expects the BPH to be supplied as a Prolog term, such as the following one:

```
node(
  name:mpop,
  natural_name:"Manage product order process",
  aggregate_function:sum_qos,
  trust_level:1,
  cost:20,
  qos:MPOP_ET,
  children:[]
)
```

Note the unbound qos variable to be constrained at a later moment. cost represents the generic cost variable depending on the scenario. In our case, cost represents the execution time of the node. aggregate_function is the predicate that is used by apply_fun to generate a constraint out of the nodes children.

The implementation walks the BPH in postorder fashion – the predicate cspec_children is applied before apply_fun. This means that walking the tree is of linear complexity. Under the reasonable assumption that the aggregation function is of linear complexity as well, the same is true for the application of cspec.

The prototype mainly exposes two predicates, one for convenient and one for programming use:

[5] http://www.eclipse-clp.org/.
[6] http://eclipse-clp.org/doc/libman/libman016.html.

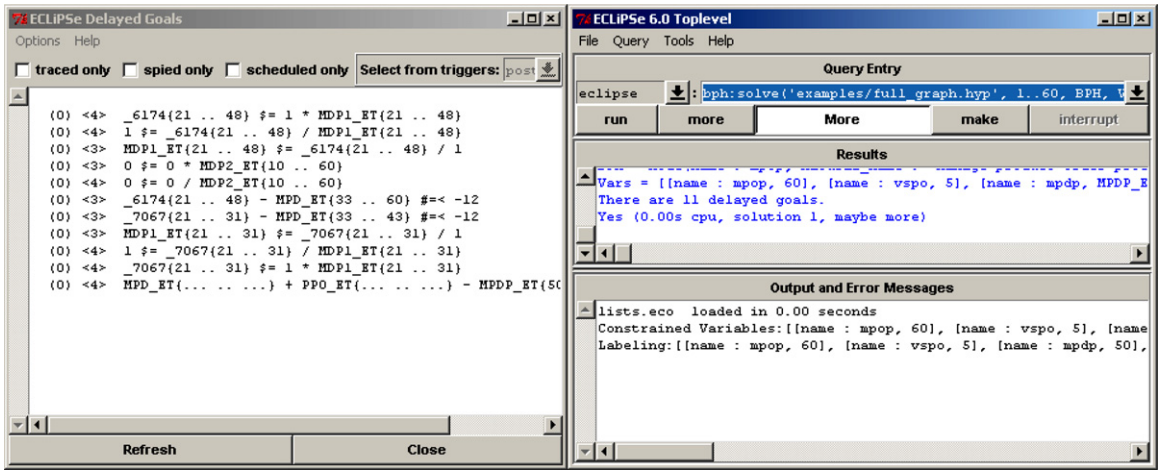

Fig. 9. tkeclipse: Main Window and Outstanding Constraints.

builds the constraint system for a given BPH and a Quality of Service domain. It then prints all constrained variables. After that, a set of possible values for those variables is computed (labeling) and printed as well.
only solves the problem of building the constraint system but does not label. It provides the user with the constrained BPH data structure as well as the list of constraint variables present in the BPH as a flat list. This gives the ability to further modify the tree and to inspect the reduced constraint system.

The easiest way to interact with the system is through the `tkeclipse` interface that is provided with ECLiPSe shown in Fig. 9. `tkeclipse` allows to directly inspect the constraint system computed and provide a good way of handling and compiling ECLiPSe modules. It requires some knowledge about Prolog but in turn provides a good way to follow the execution of the algorithm.

7. Discussion and future work

The SLAs an enterprise has with its service providers must support its business goals insofar as possible. Establishing a Service Level Agreement that favors the business objectives requires significant commitment of resources from the company side. Therefore any automation and support that can be obtained for this task is of a great benefit for the enterprise.

The presented work proposes the BP&SLA methodology for designing service-based BPs with related SLAs. The proposal fills the gap that exists between the informally specified early business requirements the user provides and the executable BP. The idea is to enrich business processes with Service Level Agreements which are desirable for the enterprise in order to achieve its business objectives. As the activities regarding assignment of responsibilities on BPs need to be carefully considered from the security point of view, the proposed methodology focuses on security and trust aspects. The framework supports the Secure BPEL language that allows for secure BPs specification. The methodology is supported by an implementation of its main algorithms based on a constraint satisfaction approach. We have used the ECLiPSe framework and the IC Hybrid Domain Solver.

The methodology proposed combines user's manual and interactive activities with automatic steps. Especially in the initial phases, the user, supported by design tools, provides the initial models. From there on, only minimal interaction is necessary to refine the models eliminating ambiguities and then guide their transformation into executable processes. In fact, the last phase is completely automatized. The initial price to pay in manual work, rewards in the final steps when having processes with SLA ready for execution. Though not experimentally confirmed, we are optimistic that our methodology will have positive evaluations similar to those obtained for Tropos, see, e.g., Estrada et al. (2006); Bresciani et al. (2004) as the first phases of BP&SLA are derived from Tropos approach. In fact, relying on portable and interoperable Web service technologies may be even a further improvement on the usability of the proposed methodology.

Looking specifically at Web service technologies, the proposed methodology builds on the following two mainstream standards: (1) WS-BPEL used to express the hierarchy of BPs, and (2) WS-Agreement used to express the SLAs. Additionally, Secure BPEL, a dialect of WS-BPEL, is used in the proposal for the functional parts abstracting away low level implementation details from WS-Security, WS-Trust and WS-Federation standards. In the transformation process from early requirements to BPEL the internal representation of the intermediate products of the methodology does not follow standard formats, but rather defines its own intermediate structures such as BP hypergraph and hierarchy of BPs. This can be improved by resorting to widely adopted standards such as BPMN (White, 2009). In other worlds, it is possible to refine Secure Tropos diagrams into BPMN diagrams and then to refine the latter into BPEL code. The last step can be done automatically, as proposed in Ouyang et al. (2007). Such a move to the BPMN standard would not change the core of the methodology, but would allow for a better community acceptance and control over the engineering process by the BP expert.

Open to future a investigation is the consideration of multi-requirement analysis which calls for the identification of a decision-making function that selects the preferred set of attributes.

Acknowledgments

This work has been partly supported by the IST-FP6-IP-SERENITY and IST-FP6-IP-C@R projects. The authors thank Fabio Massacci for comments on a previous version of this paper, Fabio Casati and Artsiom Yautsiukhin for fruitful discussion. Ganna Frankova thanks the University of Groningen for hosting her while part of the presented research was performed.

Appendix A.

Algorithm 1. Business Process Hypergraph Construction (BPHC)

```
BPHC (ST, actor, goal, QoS)
begin
  if goal is not a leaf goal
    currentNode = node (goal)
    for each children in AND
      nodei = BPHC (ST, actor, childGoal, QoS)
      interactWithUser (sequence | parallel)
      if sequence
        addHyperArcForAll (nodei[...], currentNode, sequence)
      if parallel
        addHyperArcForAll (nodei[...], currentNode, flow)
    end for
    for each children in OR
      interactWithUser (non deterministic choice | design choice)
      if non deterministic choice
        nodei = BPHC (ST, actor, childGoal, QoS)
        addHyperArcForAll (nodei[...], currentNode, switch)
      end if
      if design choice
        nodei = BPHC (ST, actor, childGoal, QoS)
        addHyperArc (nodei, currentNode)
      end if
    end for
    for each delegated child
        nodei = BPHC (ST, actor, childGoal, QoS)
        addHyperArc (nodei, currentNode)
    end for
    if trust dependency
      trustLevel(currentNode) = 1
      for each children
        trustLevel(childNode) = 1
    else
      trustLevel(currentNode) = 0
      for each children
        trustLevel(childNode) = 0
  return currentNode
  end if
  if goal is a leaf goal
    return node (goal)
  end if
end
```

The `addHyperArc(sourceNode,targetNode)` function is used to add one hyperarc in the BP hypergraph from a single source node to the target node.

The `addHyperArcForAll(sourceSetOfNodes,targetNode, aggregationFunction)` function adds one hyperarc in the BP hypergraph from a source set of nodes, i.e., `nodei[...]`, to the target node. Where the `aggregationFunction` parameter is a vector of aggregation functions $\varphi = [\varphi_1\langle N, t\rangle, \ldots, \varphi_n\langle N, t\rangle]$ assigned to the BP hyperarc.

The `interactWithUser(option1 −... − optionk)` function supports the interaction with the users with the aim to decide which structural activity to apply to for a particular goal decomposition. The users determine the proper structural activity based on the proposed options where the only one option has to be selected.

Appendix B.

Algorithm 2. Constraint System Building (CSPEC)

```
CSPEC (BPH, node, CSP, QoSDomain)
begin
  if node is not a leaf node
    addToCSP (Var node ∈ QoSDomain)
    if decomposition = AND
      for all nodes
        expr = expression (nodes, flow/sequence)
      end for
      addToCSP (Var node = expr)
    end if
    if decomposition = OR
      for all nodes
        if non deterministic choice
          expr = exression (nodes, switch)
        end if
        if design choice
          expr = expression (node, mult xi)
          where xi = 0 or 1 and sum(xi) = 1
        end if
      end for
      addToCSP (Var node = expr)
    end if
  end if
  for every node
    CSPEC (BPH, node, CSP, QoSDomain)
  end for
  if node is a leaf node
    addToCSP (Var node ∈ QoSDomain)
  end if
end
```

References

Aiello, M., Frankova, G., Malfatti, D., December 2005. What's in an agreement? An analysis and an extension of WS-Agreement. In: Proceedings of the Third International Conference on Service Oriented Computing (ICSOC 2005) , LNCS, vol. 3826. Springer, Amsterdam, the Netherlands, pp. 424–436.

Andrieux, A., Czajkowski, K., Dan, A., Keahey, K., Ludwig, H., Nakata, T., Pruyne, J., Rofrano, J., Tuecke, S., Xu, M., March 2007. Web Services Agreement Specification (WS-Agreement).

Asnar, Y., Giorgini, P., Massacci, F., Zannone, N., April 2007. From trust to dependability through risk analysis. In: Proceedings of the Second International Conference on Availability , Reliability and Security. IEEE Computer Society, Vienna, Austria, pp. 19–26.

Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., Mylopoulos, J., 2004. TROPOS: an agent-oriented software development methodology. Journal of Autonomous Agents and Multi-Agent Systems 8 (3), 203–236.

Cappiello, C., Comuzzi, M., Plebani, P., June 2007. On automated generation of web service level agreements. In: Proceedings of the 19th International Conference on Advanced Information Systems Engineering (CAiSE 2007) , LNCS 4495. Springer, Trondheim, Norway, pp. 264–278.

Castelfranchi, C., Falcone, R., July 1998. Principles of trust for MAS: cognitive anatomy, social importance and quantification. In: Proceedings of the Third International Conference on Multiagent Systems (ICMAS 1998) , IEEE Computer Society, Paris, France, pp. 72–79.

Castro, J., Kolp, M., Mylopoulos, J., September 2002. Towards requirements-driven information systems engineering: the Tropos project. Information Systems 27 (6), 365–389.

de la Vara, J., Sánchez, J., Pastor, O., June 2008. Business process modelling and purpose analysis for requirements analysis of information systems. In: Proceedings of the 20th International Conference on Advanced Information Systems Engineering (CAiSE 2008) , LNCS 5074. Springer, Montpellier, France, pp. 213–227.

Distante, D., Rossi, G., Canfora, G., Tilley, S., 2007. A comprehensive design model for integrating business processes in web applications. International Journal of Web Engineering and Technology 3 (1), 43–72.

Doerflinger, J., Friedland, C., Merz, C., De Louw, R., September 2009. Requirements of a mobile procurement framework for rural South Africa. In: Proceedings of the Sixth International Conference on Mobile Technology , Application and Systems (Mobility 2009). ACM, Nice, France, pp. 1–4.

Estrada, H., Rebollar, A., Pastor, O., Mylopoulos, J., 2006. An empirical evaluation of the i* framework in a model-based software generation environment. In: Dubois, E., Pohl, K. (Eds.), Advanced Information Systems Engineering. Lecture Notes in Computer Science, vol. 4001. Springer, pp. 513–527.

Falcone, R., Castelfranchi, C., 2001. Trust and Deception in Virtual Societies. Kluwer, Ch. Social Trust: A Cognitive Approach, pp. 55–99.

Frankova, G., Malfatti, D., Aiello, M., 2006. Semantics and extensions of WS-Agreement. Journal of Software 1 (1), 23–31.

Frankova, G., Massacci, F., Sèguran, M., July–August 2007. From early requirements analysis towards secure workflows. In: Proceedings of the Joint iTrust and PST Conferences on Privacy, Trust Management and Security (IFIPTM 2008) , IFIP, vol. 238. Springer, Moncton, New Brunswick, Canada, pp. 407–410.

Frankova, G., Yautsiukhin, A., June 2007. Service and Protection Level Agreements for Business Processes, European Young Researchers Workshop on Service Oriented Computing.

Frankova, G., Yautsiukhin, A., Séguran, M., June 2007b. From Early Requirements to Business Processes with Service Level Agreements. Tech. Re DIT-07–037, University of Trento.

Fuxman, A., Liu, L., Mylopoulos, j., Pistore, M., Roveri, M., Traverso, P., May 2004. Specifying and analyzing early requirements in Tropos. Requirements Engineering 9 (2), 132–150.

Gambetta, D., 1990. Trust: Making and Breaking Cooperative Relations. Basil Blackwell, Oxford, Ch. Can We Trust?, pp. 213–238.

Giorgini, P., Massacci, F., Mylopoulos, J., Zannone, N., October 2006. Requirements engineering for trust management: model, methodology, and reasoning. International Journal of Information Security 5 (4), 257–274.

Grandison, T., Sloman, M., October 2002. Specifying and analysing trust for internet applications. In: Proceedings of the Second IFIP Conference on Towards The Knowledge Society: E-Commerce , E-Business, E-Government. Kluwer, Lisbon, Portugal, pp. 145–157.

Grandison, T., Sloman, M., May 2003. Trust management tools for internet applications. In: Proceedings of the First International Conference on Trust Management , LNCS, vol. 2692. Springer, Iraklion, Greece, pp. 91–107.

How to Series, March 2005. SLA: Getting It Right – An Enterprise's Business Objectives Should Form the Fundamental Basis of an SLA. Voice&Data.

Jaeger, M., Rojec-Goldmann, G., Mühl, G., March–April 2005. QoS aggregation in web service compositions. In: Proceedings of the Seventh IEEE International Conference on e-Technology e-Commerce and e-Service (EEE) , IEEE Computer Society, Hong Kong, China, pp. 181–185.

Kazhamiakin, R., Pistore, M., Roveri, M., September 2004. A framework for integrating business processes and business requirements. In: Proceeding of the Eighth International IEEE Enterprise Distributed Object Computing Conference (EDOC 2004) , IEEE Computer Society, Monterey, California, USA, pp. 9–20.

Lamsweerde, A.V., 2001. Goal-oriented requirements engineering: a guided tour. In: 5th IEEE International Symposium on Requirements Engineering (RE 2001) , 27–31 August 2001, Toronto, Canada. IEEE Computer Society, p. 249.

Lapouchnian, A., Yu, Y., Mylopoulos, J., September 2007. Requirements-driven design and configuration management of business processes. In: Proceeding of the Fifth International Conference on Business Process Management (BPM 2007) , LNCS 4714. Springer, Brisbane, Australia, pp. 246–261.

Lau, D., Mylopoulos, J., June 2004. Designing web services with Tropos. In: Proceedings of the Second IEEE International Conference on Web Services (ICWS 2004) , IEEE Computer Society, San Diego, California, USA, pp. 306–315.

Ludwig, H., Keller, A., Dan, A., King, P., Franck, R., January 2003. Web Service Level Agreement (WSLA) Language Specification. Version 1.0.

Ludwig, H., Dan, A., Kearney, R., November 2004. CREMONA: an architecture and library for creation and monitoring of WS-Agreements. In: Proceedings of the Second International Conference on Service-Oriented Computing (ICSOC 2004) , ACM Press, New York, NY, USA, pp. 65–74.

Massacci, F., Mylopoulos, J., Zannone, N., 2007. An ontology for secure socio-technical systems. In: Handbook of Ontologies for Business Interaction. IDEA , pp. 188–207.

McKnight, D.H., Chervany, N.L., 1996. The Meanings of Trust. Tech. Rep. W 96–04, University of Minnesota, MIS Research Center.

Merz, C., August 2010. ICT for the Next Five Billion People, vol. VIII. Springer, Ch. Incubating Micro Enterprises in Rural South Africa – The Use Case of Virtual Buying Cooperatives, pp. 35–45.

Molina-Jimenez, C., Pruyne, J., van Moorsel, A., 2005. The role of agreements in IT management software. In: Architecting Dependable Systems III , pp. 36–58.

Ouyang, C., Dumas, M., ter Hofstede, A., van der Aalst, W., 2007. Pattern-based translation of BPMN process models to BPEL web services. International Journal of Web Services Research 5 (1), 42–62.

Papazoglou, M., Yang, J., August 2002. Design methodology for web services and business processes. In: Proceedings of the Third International Workshop on Technologies for E-Services (TES 2002) , LNCS 2444. Springer, Hong Kong, China, pp. 54–64.

Papazoglou, M.p., Georgakopoulos, d., 2003. Service oriented computing. Communications of the ACM 46 (10).

Penserini, L., Perini, A., Susi, A., Mylopoulos, J., June 2006. From stakeholder intentions to software agent implementations. In: Proceeding of the 18th International Conference on Advanced Information Systems Engineering (CAiSe 2006) , LNCS 4001. Springer, Luxembourg, Luxembourg, pp. 465–479.

Rensburg, J., Veldsman, A., Jenkins, M., 2008. From technologists to social enterprise developers: our journey as ICT for development practitioners in Southern Africa. Information Technology for Development 14 (1), 76–89.

Rolland, C., Grosz, G., Kla, R., 1999. Experience with goal-scenario coupling in requirements engineering. In: 4th IEEE International Symposium on Requirements Engineering (RE '99) , 7–11 June 1999, Limerick, Ireland. IEEE Computer Society, p. 74.

Rolland, C., Kirsch-Pinheiro, M., Souveyet, C., 2010. An intentional approach to service engineering. IEEE Transactions on Services Computing 3 (4), 292–305.

Sèguran, M., Hèbert, C., Frankova, G., November 2008. Secure workflow development from early requirements analysis. In: Proceedings of the Sixth IEEE European Conference on Web Services (ECOWS 2008) , IEEE Computer Society, Dublin, Ireland, pp. 125–134.

Tsang, E., 1995. Foundations of Constraint Satisfaction. Academic Press, London.

White, S., January 2009. Business Process Modeling Notation (BPMN), Version 1.2.

Ganna Frankova is postdoctoral researcher at the University of Trento, Italy. She hold a PhD degree in Information Technology and Telecommunications from the University of Trento in 2010. She received the Bachelor's and Master's degrees in Computer Science from the Dnepropetrovsk State University, Ukraine, in 2003 and 2004, respectively. Her research interests lie in the fields of Service Oriented Computing: quality of service, Service Level Agreement, trust and security issues for Web services, Web services-based business processes modeling.

Magali Séguran is a researcher at SAP Labs France Sophia-Antipolis in the Security and Trust research program since 2006. She holds a Master degree in computer science from the University of Nice in 1997 and a Doctoral degree in computer science in 2003 from the University of Lyon where she worked as associated researcher. She was involved in the FP7 European Project SERENITY (System Engineering For Security and Dependability). Currently she is contributing to the TAS3 (Trusted Architecture for Securely Shared Services) project with the following research areas: requirements engineering, trust and security for SOA, business processes.

Florian Gilcher is a student at the University of Applied Sciences Mannheim, Germany and the Memorial University, St. John's, Canada. In 2008 he was involved in several development projects at SAP Research France. His main interests are the theoretical and practical problems of HTTP communication.

Slim Trabelsi is a researcher at SAP Labs France Sophia-Antipolis in the Security and Trust research program since 2008. He received his PhD degree in security in ubiquitous computing from the EURECOM engineering school and the Ecole Nationale Superieure des Telecommunications Paris, France in 2008. He received his Master degree in Networks and communications from the university Pierre et Marie Curie Paris 6, France in 2004. He is currently involved in two FP7 European projects PRIMELIFE and TAS3. His research interests are related to security in SOA and particularly in service discovery, to privacy and data anonymization, and Trust and reputation models.

Jörg Dörflinger received his diploma in Communication and Media Engineering from the University of Applied Sciences Offenburg in 2007. Since 2007 he is enrolled in the PhD program at SAP Research Germany and involved in several research projects with the main research focus on technologies for emerging economies. His research interests lie in the field of ICT4D and Human Computer Interaction.

Marco Aiello is professor of Distributed Information Systems at the University of Groningen where he leads the Distributed Systems unit and coordinates several EU and Dutch National projects on Pervasive Systems, Smart Energy and eGovernment. Before he was a Lise Meitner fellow at the Technical University of Vienna, and assistant professor at the University of Trento. He holds a PhD from the University of Amsterdam and a MSc in Engineering from the University of Rome La Sapienza cum Laude.